536651

Reinforcement Learning in Distributed Domains: Beyond Team Games

David H. Wolpert
NASA Ames Research Center
Moffett Field, CA 94035
dhw@ptolemy.arc.nasa.gov

Joseph Sill
Ripfire, Inc.
San Fransisco, CA 94102
joe@ripfire.com

Kagan Tumer
NASA Ames Research Center
Moffett Field, CA 94035
kagan@ptolemy.arc.nasa.gov

Abstract

Distributed search algorithms are crucial in dealing with large optimization problem, particularly when a centralized approach is not only impractical but infeasible. Many machine learning concepts have been applied to search algorithms in order to improve their effectiveness [4, 13, 19]. In this article we present an algorithm that blends Reinforcement Learning (RL) and hill climbing directly, by using the RL signal to guide the exploration step of a hill climbing algorithm. We apply this algorithm to the domain of a constellations of communication satellites where the goal is to minimize the loss of importance weighted data. We introduce the concept of "ghost" traffic, where correctly setting this traffic induces the satellites to act to optimize the world utility. Our results indicated that the bi-utility search introduced in this paper outperforms both traditional hill climbing algorithms and distributed RL approaches such as team games.

1 Introduction

Many NASA projects under consideration involve constellations of data-communication relay satellites. In such a constellation, each satellite receives some amount of data at each time step (e.g., uplink from Earth or Mars, depending on where they are orbiting), and needs to relay this data back to an ultimate destination (e.g., Earth) with minimal loss. Although each satellite may have a direct link to the ultimate destination at particular times, because of various limitations (e.g., storage, power, bandwidth), it may still be preferable to route the data across other satellites in the networks. Furthermore, the data is likely to have different levels of importance, and the routing algorithm needs to account for that possibility. For such problems, a suitable utility function to minimize is the total loss of importance weighted data across the network.

in a single time step, respectively. We represent Earth as a "special" satellite s_0 with $c_0 = \infty$. At each time step t, new data y_{ijt} of importance j is introduced to the system at satellite i (this corresponds to the uploading of data from e.g. the surface of Mars). We sum the y_{ijt}'s over j and add this total to the total amount of data sent to satellite i from all other satellites to give the total influx of data x_{it} at this time step. If this total is greater than the available storage capacity (given by $c_i - r_{it}$, where r_{it} is the amount of unsent data on the disk left over from the previous time step), then the difference between these two numbers is the amount of data lost at satellite i at t. We assume that the same proportion of data is dropped for each importance level, since once the disk is full the satellite is unable to examine any data sent to it and determine its priority. Define l_{ijt} to be the amount of data of importance j dropped at satellite i. Define the cost of dropping data of importance j as w_j. Then the objective function we wish to maximized is:

$$G = 1 - \frac{\sum_t \sum_i \sum_j w_j l_{ijt}}{\sum_t \sum_i \sum_j w_j y_{ijt}} , \tag{1}$$

the importance-weighted percentage of data delivered to Earth (i.e., **not** dropped) by the system as a whole.

The base routing algorithm has rough parallels with the shortest path style routing algorithms commonly used in internet routing [2, 3, 7]. Each satellite i evaluates a potential decision to send to satellite k by estimating the "headroom" of the optimal path to Earth beginning at k. The headroom represents the available room for additional data, given the available disk room on each satellite and the capacity of each link between them (the headroom replaces the traditional "delay" concept in data routing). Denote a path by a sequence of satellites $\rho \equiv s_{k_1}, \ldots, s_{k_p}$, where s_{k_1} represents the originating satellite and s_{k_p} is the satellite which ultimately sends the packet to Earth. Let the current amount of data being stored at satellite i be v_i. Then the headroom $H(\rho)$ of a path ρ is given by:

$$H(\rho) = \min_{q=1}^{p-1} (\min(b_{s_{k_q}, s_{k_{q+1}}}, c_{s_{k_{q+1}}} - v_{s_{k_{q+1}}})) . \tag{2}$$

The presumption is that a path with high headroom should be favored over one with low headroom, since the likelihood of data being dropped is lower (just a path with low expected delays is favored over a path with high delays in traditional data routing). Note that in a real system, a particular satellite i would not have access to the precise storage amounts $v_j, j \neq i$ at the current instant in time. Hence, the headroom values would have to be estimated by satellites (in ways similar to how delays are estimated in traditional data routing). In these experiments, we supply each satellite with the v_j's. Because we are interested in how to improve the performance of the base algorithm, the estimation of the v_j's would introduce a systematic error that would not affect the ranking of the algorithms discussed below.

It is straightforward to calculate the maximum headroom path from each satellite to Earth using a version of Dijsktra's shortest path algorithm [3, 9]. One very simple strategy would be for each satellite to send all of its data to the first satellite along the maximum headroom path to Earth. This technique may not be optimal, however, if the amount of data the satellite sends is large relative to the headroom of the maximum headroom path. In particular, if the amount of data sent is larger than the headroom, then it is quite likely that some of that data will be dropped, unless more headroom happens to "clear up" fortuitously. It therefore seems wiser to split the data up and send it through more than one satellite.

Because of the utility function to optimize, traditional routing algorithms (e.g., shortest path algorithms [2, 3, 9]) are ill-suited for this problem. As such we develop a baseline routing algorithm that addresses the needs of this utility function. We then introduce the concept of "ghost" traffic, which distorts how the data traffic appears to the satellites, and induces them to take actions that are beneficial to the system as a whole. The objective then reduces to setting this ghost traffic properly. Although one may use search algorithms such as simulated annealing [1, 5, 8] to set the ghost traffic, this problem naturally lends itself to a Reinforcement Learning (RL) [11, 16] approach.

The use of Reinforcement Learning (RL) has proved successful in a multitude of optimization problems [4, 6, 12, 13, 14, 18, 19]. Furthermore, in a different context, distributed RL where many agents independently attempt to maximize a world utility (e.g. "team games") has been successfully used to solve large decentralized problems [6, 10, 13, 15]. In this article we discuss a combination of these two concepts into a "guided" search algorithm that uses distributed RL to improve upon both traditional team game solutions and traditional search algorithms.

In this article we present a search algorithm that combines team games and hill climbing and apply it to minimizing loss of information on a constellation of communication satellites. In Section 2 we detail the problem domain, and introduce the concept of "ghost" traffic. Then, in Section ??, we present the bi-utility search algorithm, and discuss both its motivation and applicability. Finally, in Section 3 we present experimental results demonstrating the superiority of the bi-utility search algorithm over both team games, simulated annealing and hill climbing algorithms.

2 Constellations of Satellites

One of the key challenges in the design of constellations of communication satellite networks is the development of routing algorithms which minimize the amount of data lost by the system as a whole. In such constellations, each satellite's storage capacity, downlink bandwidth and available power is certain be limited. If a satellite's disk becomes full, then any further incoming data will be lost. This predicament can potentially be avoided if the satellite clears storage space by sending some of its data to a neighboring satellite with more room on its disk and/or a larger bandwidth link to Earth. In general, in order for the entire constellation to minimize loss, data packets may need to be routed through several satellites before ultimately being delivered to Earth. Further complicating this task is the fact that different data packets may have different levels of importance. Routing decisions should reflect this variability in priorities.

This task can be characterized by a well-specified global objective function (minimize importance-weighted amount of data dropped), and yet it is fundamentally decentralized in nature, since it is infeasible to disseminate routing decisions from a single, centralized source. Furthermore, some sort of adaptivity is clearly required, since the complexity and potential non-stationarity of the problem is certain to make any hand-designed scheme both brittle and undoubtedly sub-optimal.

2.1 Model Description

We model the evolution of the system as a sequence of discrete time steps. A constellation of satellites is specified by the following set of parameters: Each satellite s_i has a storage capacity c_i and a link capacity (bandwidth) b_{ik} to each other satellite, where c_i and b_{ik} are real numbers indicating the amount of data the satellite can store and the amount of data the satellite can transmit to each other satellite

2.2 Baseline Routing Algorithm

Let H_i be the headroom of the max headroom path from satellite i to Earth. Let H_{ij} be the headroom of the optimal path originating at satellite i and with the first hop being to satellite j. So H_{ij} is given by

$$H_{ij} = min(b_{ij}, \mathcal{H}_j)$$

The satellite at which data originates does not decide on the full path to Earth taken by its packets; it simply decides on the first hop in the path and sends each chunk to the appropriate satellite based on the H_{ij}'s. (similarly, in traditional data routing, a router only selects the first hop along the seemingly shortest path, based on the delays). The routing is performed as follows: Let v_{ik} be the amount of data of importance k currently at satellite i, and Define

$$j^* = \text{argmax}_j H_{ij}.$$

If $H_{ij^*} > v_{ik}$, then all v_{ik} is sent to satellite j^* and H_{ij^*} is updated by subtracting v_{ik} off of the headroom estimate to reflect the fact that that much data has already been sent to that satellite. If $H_{ij^*} < v_{ik}$, then an amount H_{ij^*} is sent to j^* and H_{ij^*} is updated to equal zero.

The whole procedure is then repeated until either (1) $v_{ik} = 0$ or (2) $H_{i,j} = 0 \forall j$. If the second condition occurs before all data has been routed, then the remaining data is not sent anywhere and instead kept on the disk until the next iteration in the hopes of routing it successfully then.

The routing algorithm factors in the importance levels of different data by performing the routing for the highest importance data first, and then successively moving down to lower and lower importance levels until either all the data has been routed or all the headroom estimates are zero.

2.3 Routing with "Ghost" Algorithm

The routing scheme discussed above shares common features with the Shortest Path Algorithms (SPAs) for routing [3, 9], though it significantly outperforms them in this domain. This is not surprising as traditional shortest path algorithms do not handle data of varying importance well (unless quality of service considerations are included). Still, while it performs respectably on its own, it is susceptible to the same phenomena that hamper SPAs in traditional routing [17]: the satellites do not explicitly act to optimize G, and can therefore potentially work at cross-purposes.

We now introduce the concept of "ghost" traffic to alleviate these concerns, and discuss several techniques for setting the levels of this traffic. Let us introduce distortions to the headroom estimates H_{ij} given by: δ_{ij}. In other words, we set

$$H_{ij} = min(b_{ij}, \mathcal{H}_j) + \delta_{ij}$$

and then perform the routing according to the perturbed headroom estimates. The $\delta'_{ij}s$ are free parameters to be determined via an optimization algorithm, and because their effects on the headroom is the same as that of actual data, we call them "ghost" traffic. Our goal then is to find a set of δ's such that the performance of the system is substantially improved as compared to the performance of the base, deterministic routing strategy.

If we collect all the δ_{ij}'s into a single vector $\vec{\delta}$, it becomes clear that what we are faced with is a multidimensional optimization problem. Each particular choice of $\vec{\delta}$ will yield a particular global performance $G(\vec{\delta})$.

3 Experimental Results

In this section we compare the performance of bi-utility search to both that of team games, and that of hill climbing in the problem of setting the "ghost" traffic level to to minimize the provided world utility. In this context we associate an "agent" with each δ (i.e., amount of ghost traffic on a particular link). In this section we present the results of the following 5 algorithms:

- **Baseline:** Algorithm outlined in Section 2.2; No learning.
- **Team Games:** Each agent uses RL to try to independently maximize world utility.
- **Random search:** At each step, a random δ vector is generated and that vector is probabilistically "accepted" based on the world utility it provides (simulated annealing).
- **Random Hill Climbing:** Similar to random search, but the new delta vector is constrained to be one "bin" away from the old one.
- **G-Based Hill Climbing:** Similar to random hill climbing, but the exploration step is guided by the RL estimates of the G rewards associated with each action under consideration.

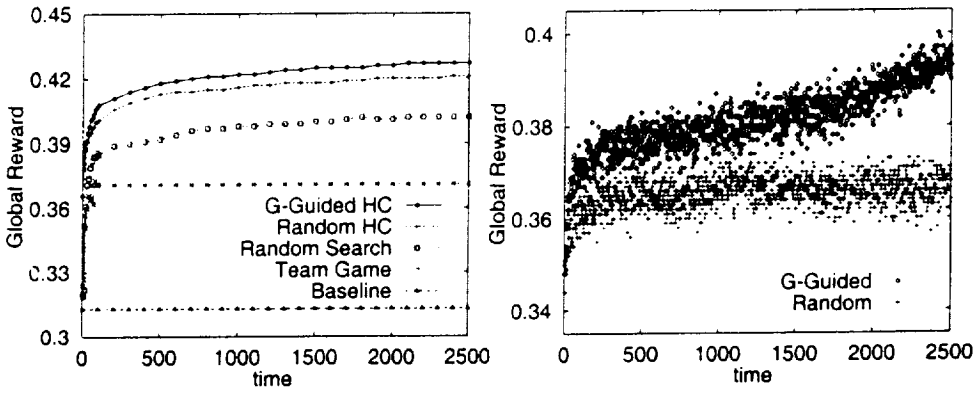

Figure 1: (a) Performance of the various algorithms in the Satellites Communication domain; (b) Exploration by Traditional and G-Guided Hill Climbing Algorithms.

Figure 1 (a) compares the performance of the five algorithms described above on a problem with 20 satellites of moderate connectivity (150 δ's), averaged over 100 runs (the resulting error bars are too small to depict). The use of a team game approach provides an improvement over the baseline algorithm. However, random search shows that the team game strategy fails to take full advantage of the possible gains. The hill-climbing results provide another significant jump over the gains achieved through simulated annealing. While both hill climbing algorithms are superior to team games, the G-guided search which incorporates components of both team games and hill climbing clearly outperforms traditional hill-climbing.

Figure 1 (b) provides an analysis on the reasons why G-guided exploration is superior to traditional hill climbing. The plots show the states generated by the exploration steps of the two hill climbing algorithms. The exploration guided by G clearly provides better states, which directly translates into higher performance. Furthermore, the quality of the steps generated by the random hill climbing algorithm's exploration show no improvement over time. In contrast, the exploration steps generated by the G-guided hill climbing algorithm show a clear improvement.

4 Conclusion

Distributed search algorithms are crucial in dealing with large optimization problem, particularly when a centralized approach is not only impractical but infeasible. Team game solutions to this problem provide some relief, but often fall short of the potential gains (e.g., due to a lack of coordination among agents). Local search algorithms provide another approach to this problem, but also fail to maximize the gains, in that they do not exploit any knowledge acquired from the previously explored states.

In this article we present an algorithm that blends these two concepts into a global utility directed search algorithm. We apply this algorithm to a distributed information routing problem where the global objective is to minimize loss of importance-weighted data. By accurately setting the "ghost" traffic introduced to prod the agents to act to optimize world utility, this algorithm outperforms both traditional hill climbing algorithms and team game algorithms.

References

[1] E. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing.* John Wiley and Sons, 1989.

[2] R. E. Bellman. On a routing problem. *Quarterly of applied mathematics*, 16:87–90, 1958.

[3] D. Bertsekas and R. Gallager. *Data Networks.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[4] J. A. Boyan and A. W. Littman. Learning evaluation functions for global optimization and boolean satisfiability. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence.* AAAI Press, 1998.

[5] O. Catoni. Solving scheduling problems by simulated annealing. *SIAM Journal on Control and Optimization*, 36(5):1539–1575, 1958.

[6] R. H. Crites and A. G. Barto. Improving elevator performance using reinforcement learning. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems - 8*, pages 1017–1023. MIT Press, 1996.

[7] N. Deo and C. Pang. Shortest path algorithms: Taxonomy and annotation. *Networks*, 14:275–323, 1984.

[8] R. Diekmann, R. Luling, and J. Simon. Problem independent distributed simulated annealing and its applications. In *Applied Simulated Annealing*, pages 17–44. Springer, 1993.

[9] E. Dijkstra. A note on two problems in connection with graphs. *Numeriche Mathematics*, 1(269-171), 1959.

[10] J. Hu and M. P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 242–250, June 1998.

[11] L. P. Kaelbing, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[12] M. Kearns and D. Koller. Efficient reinforcement learning in factored mdps. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 740–747, 1999.

[13] M. L. Littman and J. Boyan. A distributed reinforcement learning scheme for network routing. In *Proceedings of the 1993 International Workshop on Applications of Neural Networks to Telecommunications*, pages 45–51, 1993.

[14] R. Moll, A. G. Barto, and T. J. Perkins. Learning instance-independent value functions to enhance local search. In *Advances in Neural Information Processing Systems - 11*, pages 1017–1023. MIT Press, 1999.

[15] T. Sandholm and R. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems*, 37:147–166, 1995.

[16] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[17] K. Tumer and D. H. Wolpert. Collective intelligence and Braess' paradox. In *Proceedings of the Seventeeth National Conference on Artificial Intelligence*, 2000. to appear.

[18] W. Zhang and T. G. Dietterich. A reinforcement learning approach to job-shop scheduling. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1114–1120, 1995.

[19] W. Zhang and T. G. Dietterich. Solving combinatorial optimization tasks by reinforcement learning: A general methodology applied to resource-constrained scheduling. *Journal of Artificial Intelligence Reseach*, 2000.